



---

# **Global Mobile Financial Services Partner Developer's Guide**

---

Version 0.12

## Table of Content

<b>1. Introduction to Millicom</b>	<b>5</b>
<b>2. About this guide</b>	<b>6</b>
2.1. Introduction	6
2.2. Conventions	6
2.2.1. Code and API references	6
2.2.2. Sample data	6
2.2.3. Warning	6
2.2.4. Note	6
2.3. Definitions	6
<b>3. Application integration guidelines</b>	<b>7</b>
3.1. Connectivity and communication	7
3.2. Integration steps	7
3.3. Partner Mobile Accounts	8
3.4. Session Access Token	9
3.5. System Status heartbeat signal	9
3.6. International Remittance Money Deposit	9
3.7. Payment Authorization	11
3.7.1. Payment Authorization via SMS verification code	11
3.7.2. Payment Authorization via USSD Push	15
3.8. Reverse Transaction	17
<b>4. API Specification</b>	<b>19</b>
4.2. Introduction	19
4.3. Generate Access Token Service	19
4.3.1. Request	19
4.3.2. Response	20
4.4. System Status Service	21
4.4.1. Request	21
4.4.2. Response	21
4.5. Payment Authorization service	22
4.5.1. Request	22
4.5.2. Response	24
4.5.3. Payment status callback	25
4.6. Validate MFS Account Service	28
4.6.1. Request	28
4.6.2. Response	29
4.7. Deposit Remittance Service	30
4.7.1. Request	30
4.7.2. Response	32
4.8. Reverse Transaction service	33
4.8.1. Request	33
4.8.2. Response	34
4.9. Payment Authorization Transaction Status service	35
4.9.1. Request	35
4.9.2. Response	35
4.10. Deposit Remittance Transaction Status service	38
4.10.1. Request	38
4.10.2. Response	38
<b>A. Appendices</b>	<b>41</b>
A.1. Currency Codes	41

A.2.	Country Codes .....	42
A.3.	Language Codes .....	43
A.4.	Country Calling Codes .....	44
A.5.	Amount Format Convention .....	45
A.6.	Result and error codes .....	46
A.6.1.	<i>HTTP status codes</i> .....	46
A.6.2.	<i>General error codes</i> .....	46
A.6.2.1.	<i>IP address not whitelisted</i> .....	46
A.6.2.2.	<i>Invalid Request</i> .....	46
A.6.2.3.	<i>Invalid Access Token</i> .....	47
A.6.2.4.	<i>Access Token Expired</i> .....	47
A.6.2.5.	<i>Transaction Reference ID already used</i> .....	47
A.6.3.	<i>API Permission error</i> .....	47
A.6.4.	<i>Service specific result and error codes</i> .....	47
A.6.4.1.	<i>Payment Authorization Service result codes</i> .....	49
A.6.4.2.	<i>Validate MFS Account Service result codes</i> .....	49
A.6.4.3.	<i>Deposit Remittance Service result codes</i> .....	51



## Version Control

Date/Time	Version	Changes
01/09/2014	0.1	First Draft
16/09/2014	0.2	Added Payment Authorization flow
18/09/2014	0.3	Added section on MFS accounts/wallets
22/09/2014	0.4	Updated API fields after review
29/09/2014	0.5	Added Sections 3.4, 4.2 for the heartbeat signal, Updated Section 4.4.5 payment authorization callback
6/10/2014	0.6	Update after internal review
9/10/2014	0.7	Section 3.7: Updated Payment Authorization sequence diagram with customer redirect; Section 4: Added <domain> to the URLs in the API specifications; Section 4.4: Added example System Status Service; Section 4.5.1: Updated Payment Authorization URL, added description on the fees in the Payment Auth request, Added currency code for Merchant fee; Section 4.5.2: Added Payment Authorization response with redirect URL; Section 4.5.3: Added section for Payment Status call back; Section 4.8.1: Updated Payment Authorization status response structure
15/10/2014	0.8	Section 3.7: Updated Payment Authorization screenshots to align with new UI design and updated description Section 4.7: Included Sender details in Remittance request Section 4.8: Added createdOn and completeOn fields Annex A.4.6.2: updated Deposit Remittance Error codes Included Payment Authorization error codes
27/11/2014	0.9	Minor typo updates and clarifications throughout the document; Section 3.6: updated description and flow for international remittance; Section 4.6: Changed the First name, last name fields in the Validate MFS account to be optional; Section 4.7: made the verificationRequest parameter optional and added explanation that this feature is not supported in current version, added an optional flag to send a text message; Section 4.8 and Section 4.9: Updated the transaction reference for looking up the Payment Authorization Transaction Status and Deposit Remittance Transaction Status to include the company name appended to the transaction reference ID.
28/11/2014	0.10	Added two warnings in Payment Authorization request Section 4.5.1 on the access token and transaction reference id;
12/12/2014	0.11	Added Reverse Transaction (Section 3.8 and 4.8 Reversed change in OriginPayment (Section 4.5.1) Added Amount formatting (Annex)
13/04/2015	0.12	Add new error code with description for the Cancel Transaction scenario

## 1. Introduction to Millicom

Founded in 1990 and headquartered in Luxembourg with corporate offices in Stockholm, London and Miami, Millicom's subsidiaries operate exclusively in emerging markets in Africa and Latin America. Millicom's shares are listed on the Nasdaq OMX exchange in Stockholm and Millicom's market capitalisation was SEK64 billion (\$9.9 billion) at the end of 2013.

Millicom is a leading international telecommunications and media company dedicated to emerging markets in Latin America and Africa. We set the pace by offering a range of digital lifestyle services which connect people to their world. Operating in 15 countries, with e-commerce partnerships in 22, Millicom offers innovative and customer-centric products. Millicom employs over 11,500 people and provides mobile, cable, broadband, TV and mobile financial services to over 50 million customers.

Millicom has divided its business into four strategic pillars. An overview of our progress on the strategic pillars is given as follows:

Strategic pillar	Progress in 2013 <sup>a</sup>	Comment on progress	Outlook
<b>Mobile</b> Move from volume to value	<b>Revenue in 2013</b> <b>\$4.2bn</b> <b>Growing 3%</b> on a recurring basis	Driven by innovation, Data and Value Added Services (VAS) are increasing in mobile revenue providing new sources of growth. Our unlimited music offer has been extremely successful in many markets.	Many of our African countries are gaining significant traction and should continue to do so next year.  B2B market focus in Latin America should create the next wave of growth in the region.  Data and VAS is expected to continue to increase share as we launch new products and features increasing mobile revenue.
<b>Cable &amp; Digital Media</b> Build a sizeable business	<b>Revenue in 2013</b> <b>\$446m</b> <b>10% organic growth</b>	Progress has been achieved both through organic growth and acquisitions. Our newest operations in Paraguay (further rollout after acquisition of Cablevision) and Guatemala (greenfield roll-out with some in-market consolidation) have experienced double digit growth. Costa Rica this year was negatively impacted by competition from low-end providers.	The merger with UNE in 2014 will be a major step in achieving our goal.  The launch of Tigo Sport, initially in Paraguay, and in the rest of Latin America thereafter, is a major development in our content strategy, which we believe will lead to additional opportunities.
<b>MFS</b> Creating a blockbuster	<b>Revenue in 2013</b> <b>\$79m</b> <b>98% growth</b> <b>6.3 million customers</b>	In 2013 we have seen Tanzania, Rwanda, Paraguay and El Salvador with some of the highest adoption rates in the world grow even further. New products have been launched in every market contributing to our current and future growth.	Some of our operations are introducing Mobile Financial Services and growth in penetration will drive revenue. Our product pipeline is strong and we expect new sources of revenue in MFS as the numbers of transactions increase, in particularly peer-to-peer.
<b>Online</b> Investing in growth	<b>Revenue in 2013</b> <b>\$83m</b> <b>554% growth</b>	Online has been driven mainly by e-commerce ventures with companies like Kanui and Tricae in Brazil, and Jumia in Africa with its extremely high growth rates. In parallel, many other ventures are building the path for future growth.	Latin America has shown a strong adoption of online services and should continue in this path. Our partnership with MTN in Africa Internet Holdings at the end of 2013 will push our Online ventures further in Africa, which should accelerate growth.



## 2. About this guide

### 2.1. Introduction

This reference document is intended for Millicom partners to plan, build and deploy applications that wish to connect and use the Tigo Mobile Financial Services (MFS).

The two services currently provided:

- Online payments
- International money transfers to Tigo mobile wallet.

This document covers all the implementation details concerning network connectivity, security and API specification in order to successfully integrate with the International Tigo MFS services. The step by-step guide will guarantee successful and secure integration with the services.

### 2.2. Conventions

#### 2.2.1. Code and API references

Any code or API references are specified in the Courier font: for example see the `GenerateAccessToken` API

#### 2.2.2. Sample data

Sample data such as responses and requests are shown in a separate yellow colored text box

#### 2.2.3. Warning



**Warning:** is shown in a red colored text box with an icon to catch the attention. Most warnings are security related.

#### 2.2.4. Note



A note is shown to catch attention and provide useful information and extra explanation of the applicable section

### 2.3. Definitions

Term	Definition
API	Application Programming Interface
UAT	User Acceptance Testing
JSON	JavaScript Object Notation
MFS	Mobile Financial Services
OTP	One Time PIN
SSL	Secure Sockets Layer
URI	Uniform Resource Identifier

### 3. Application integration guidelines

#### 3.1. Connectivity and communication

The diagram below shows a high level view of the international integration for the MFS services with the Tigo Mobile Operators in Latin-America and Africa. Integration is done via a central Tigo Secure Server hosted on Apigee. This centralized server takes care of access control, routing, requests to the corresponding Tigo operation and most importantly the secure handling of data. All communication on all the interfaces is encrypted.

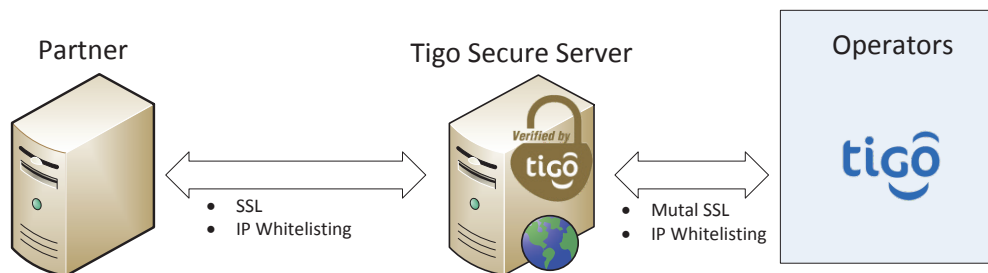


Figure 3-1: Architecture overview

Millicom Tigo Secure is hosted on Apigee in a minimum of two datacenters. This mission critical platform has up to 99.99% availability. The cloud environment provides load-balancing and failover across the multiple server instances.

All the provided services are exposed to use JSON.

Two main URIs are provided for integration:

<https://securesandbox.tigo.com/> test environment  
<https://secure.tigo.com/> production environment

All communication with the Tigo Secure Server use HTTPS / SSL to exchange information. The Payment Authorization solution is established via one-way SSL. The international remittance deposit money service requires two-way SSL and IP whitelisting.

#### 3.2. Integration steps

The following steps have to be carried out in order to integrate successfully with the Millicom Tigo Secure environment and the Mobile Financial Services:

1. Register with Millicom Tigo
2. Acquire an Apigee API Key and secret
3. Exchange SSL certificates for 2-way SSL
4. Make sure that MFS Accounts have been created in the respective countries and that account numbers and pin codes are known

5. Submit the IP address of the server(s) that will connect with Tigo Secure in order to whitelist

### 3.3. Partner Mobile Accounts

For each Millicom Tigo operation with which a partner will integrate a separate MFS Account (also called mobile wallet) has to be opened. Each account is uniquely identified with either a MSISDN or username and a PIN code. For each service call interacting with the MFS Account the correct account details have to be provided in the request for the designated country.

Opening an account the exact process depends on the country and in general involves the following steps:

- Signed NDA
- Company to provide KYC details
  - *Differs from country to country but high level is:*
  - Business Name
  - Business License
  - Tax Identification Number
  - Stated Capital
  - Contact person(s) details & ID
- Bank account details of account in local bank

Depending on the use cases and the functionality/product launched these can be broadly classified into 2 kinds of accounts:

1. Pre-Funded Account
2. Collection Account

1. **Pre-funded account**- This type of account is provided in case of integrations where the partner is required to have virtual money (e-money/local MFS currency) in advance to make transfers into the end users wallets. The local process of procuring this MFS currency (e-money) differs per operation but it usually involves depositing actual money in the local currency into the bank account designated by the Tigo operation and getting a mirror value replicated in the MFS platform (as e-money or MFS currency).

Typical products and functionalities using this type of account are Disbursements, Remittance transfers, Transfers.

The settlement process is usually agreed between both entities that governs the management of the e-money and real money

2. **Collection Account**- This type of account is provided in cases where the partner is required to collect or accumulate transfers into their accounts. The end user that has a valid account would be able to transfer e-money/MFS currency into the partner account for the intention of making payments, transfers, purchases etc.



Typical products and functionalities using this type of account are:  
Merchant Payments, Bill payments, transfers, goods purchases.

As confirmed before the settlement process defined governs the movements and transfer between the collection account and the partner's bank account.

### 3.4. Session Access Token

For each session a valid Access Token has to be requested via the `GenerateAccessToken` service (see section 4.3) using the API key and secret. This Access Token has a limited validity period. After completing a session (either successful or unsuccessful) the access token will be invalidated. The process is shown in the next sections.



**Warning:** You should never authenticate using the API Key and Secret directly from a client-side app such as a mobile app. A hacker could analyze your app and extract the credentials for malicious use even if those credentials are compiled and in binary format. [Source: Apigee]

### 3.5. System Status heartbeat signal

The system status is monitored by sending a periodic request to the Tigo Secure server. In the response the status is reported of each of the Tigo operations. A lack of response will mean the service is down caused by a network error or other failure. These events should be logged and alerted on to Tigo in order to be restored to normal operation.

### 3.6. International Remittance Money Deposit

The process to deposit money for international remittance is shown in Figure 3-2 below.

(1,2) An Access token has to be requested for the Tigo Secure Server via the `GenerateAccessToken` service (section 4.3) using the Apigee API Key and secret.

(3-6) The next optional step is to Validate the MFS Account via the `ValidateMFSAccount` service (section 4.6.1). In case no validation is done and the receiving Tigo subscriber does not have an MFS account then the next step to actually deposit the remittance will fail in which case an (optional) text message is sent suggesting the subscriber to sign up of an MFS account.

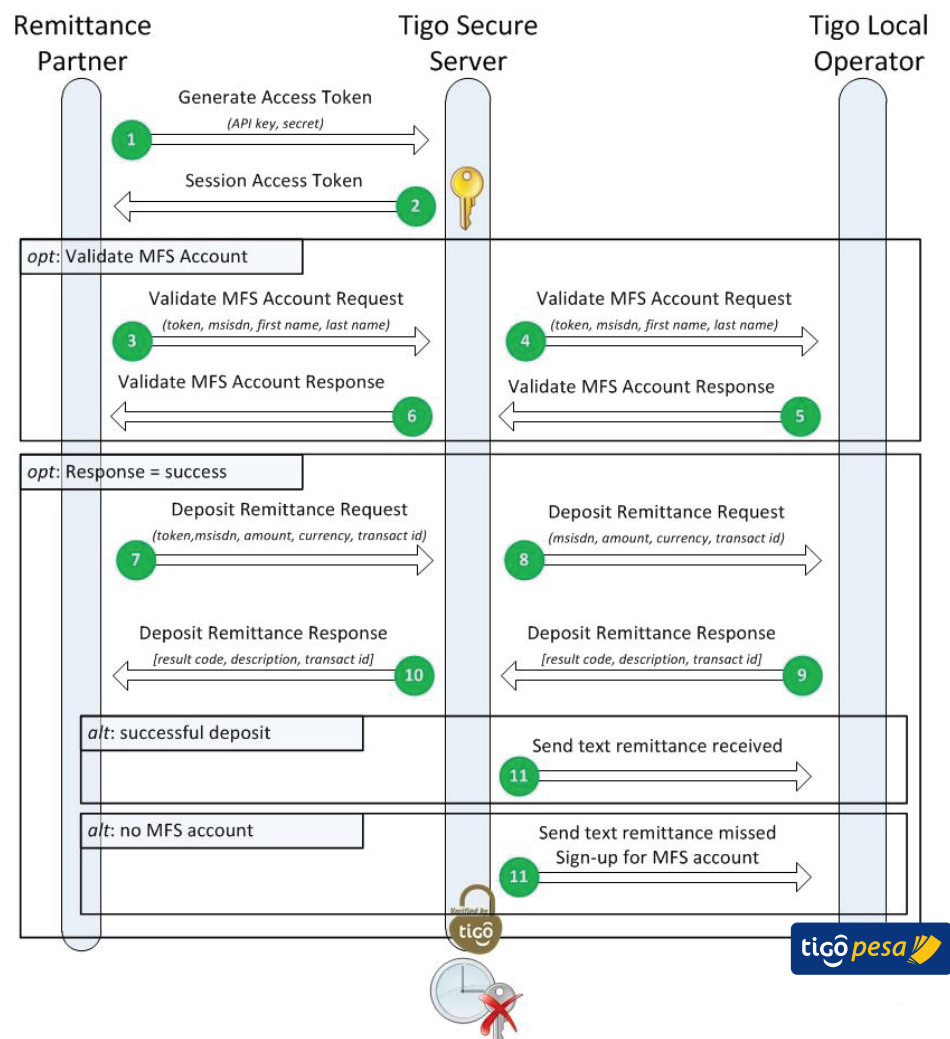


Figure 3-2: International Remittance flow

(7-10) the Remittance Partner can deposit the amount in the local currency via the `DepositRemittance` service (section 4.7.1).

(11) The receiving Tigo subscriber will receive a text message in case this is specified in the request. This text message will be generated in the following two scenarios:

- Successful deposit a text message informing the subscriber that an international remittance has been received with the amount, name of the remittance partner and optionally the name of the sender (if provided in the request)
- Unsuccessful remittance cause by the receiving subscriber not having a MFS wallet a text message informing the subscriber that an international remittance was missed with the amount, name of the remittance partner, optionally the name of the sender (if provided in the request) - and the advice to open a Tigo MFS account

The Access Token is invalidated after the expiry time as specified in the Generate Access Token Response (section 4.3.2).



### 3.7. Payment Authorization

The Payment Authorization service is based on a URI redirect whereby the actual payment verification and authentication by subscriber is entirely handled on the Tigo Secure server. The next sections show the flows of the payment authorization where the verification is done via SMS in case of the Africa region and via USSD push for LATAM.

The initial language of the Tigo Secure webpages shown is set via the language parameter in the request. It is preferable to keep the language the same as the page from which the customer is redirected. The customer has the option to select a different language on the webpage itself as well.

#### 3.7.1. Payment Authorization via SMS verification code

In the following countries the payment is authorization by sending a verification code via text message to the subscriber:

- Senegal
- Tanzania

This verification code is only valid for a limit duration of 1 minute and has to be filled in by the Customer on the Tigo Secure webpage. Besides this verification code the customer also has to provide their MFS PIN code. The flow is shown below.



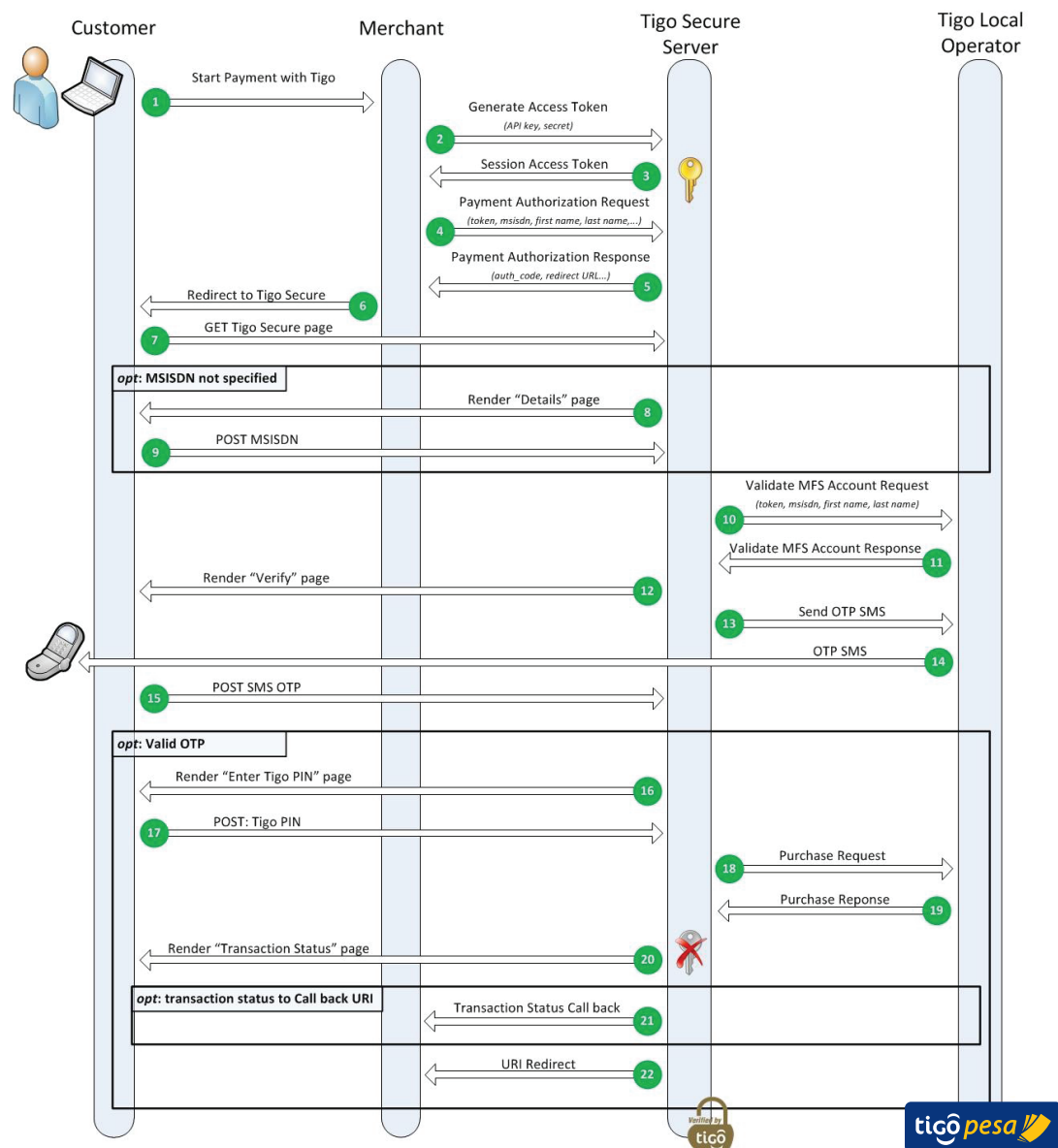


Figure 3-3: Payment Authorization flow SMS OTP

- (1) The subscriber/customer initiates a Tigo MFS payment via the Merchant.
- (2-3) An Access token is requested for the Tigo Secure Server via the `GenerateAccessToken` service (Section 4.3.1) using the Apigee API Key and secret.
- (4-8) The Payment Authorization Request is made with the necessary payment details (Section 4.5.1) this will return a re-redirect URL to the Tigo Secure Payment Authorization page to which the customer has to be redirected.



- (9) In case the MSISDN is not yet specified in the Payment Authorization request or in case the MSISDN was incorrect (non-existent) then the Customer is redirected to a page to enter the MSISDN.

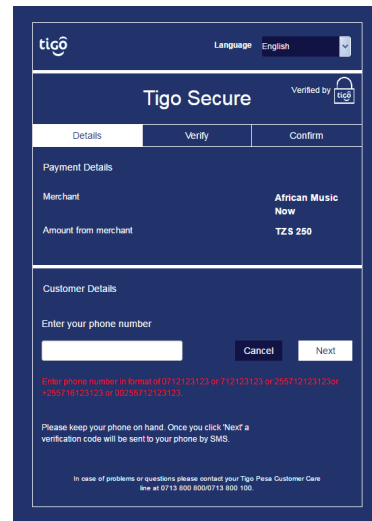


Figure 3-4: Enter verification code page

- (10-14) The MFS Account of the subscriber is validated after which the 'Verify' Page is shown. (see Figure 3-5) and a One-Time-Pin (Verification code) is sent via text message to the Tigo subscriber.

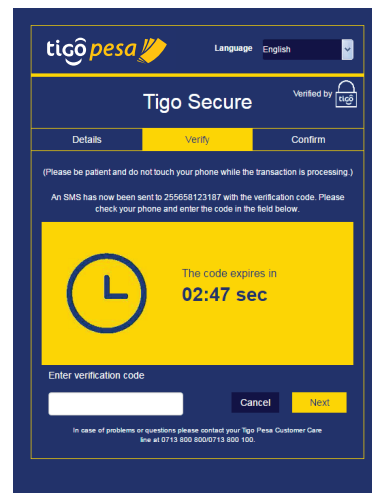


Figure 3-5: Enter verification code page

- (15 - 16) The subscriber submits the verification code and after successful verification of the code the payment overview is shown to the customer requesting the Tigo MFS Account PIN (Figure 3-6).

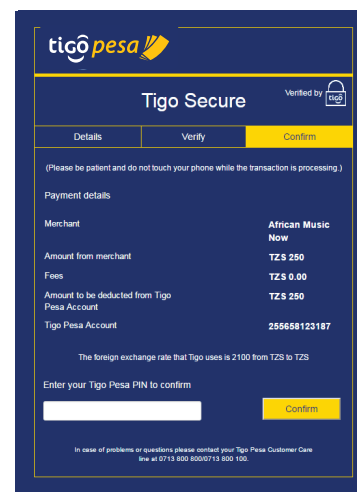


Figure 3-6: Payment overview page

- (17-19) The customer provides the MFS PIN and the purchase call to make the payment is sent.
- (20) Upon receipt of a successful purchase response the access token is invalidated and a payment result page is shown for a limited duration (Figure 3-7).

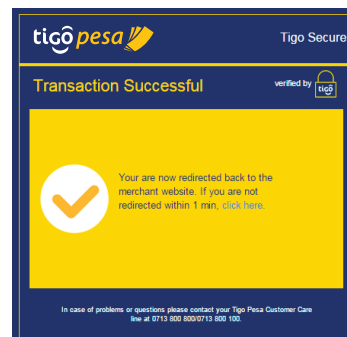


Figure 3-7: Payment result page

- (21) An optional callback URI is called with the final transaction status. This callback URI can be used in case the front-end server does not allow processing the financial transaction status.
- (22) The final redirect is done to the specified redirect URI.

The non-nominal cases for the Payment Authorization using SMS verification are shown below

### Invalid Verification Code

When the subscriber enters the incorrect verification code a warning is shown “*Invalid Verification Code. Please re enter*” with the possibility to try again. The number of attempts is limited by the expiry time of the verification code.

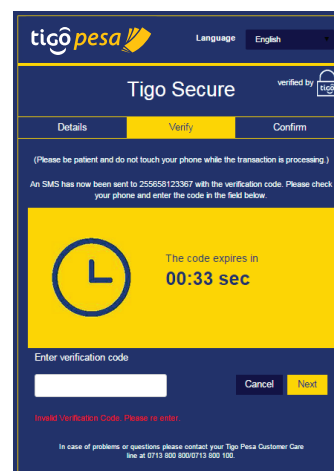


Figure 3-8: Invalid Verification Code



### Verification Code expired

In case the verification code expires a warning is shown *“The code has now expired. Please make sure you have the phone at hand and click below to resend the code.”* with the option to resend a verification code. Resending the verification code is limited to 3 times.

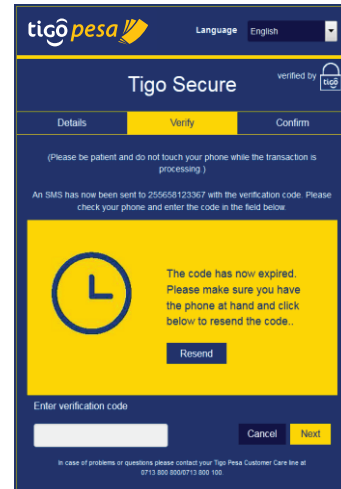


Figure 3-9: verification code expired

### Incorrect PIN code

In case the subscriber enters an incorrect PIN code a warning is displayed *“PIN was not valid. Please enter the PIN again.”* The subscriber has three attempts to re-try. After that the subscriber account will get blocked.

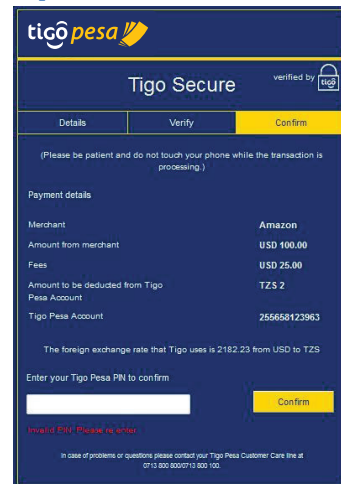


Figure 3-10: incorrect PIN code

### 3.7.2. Payment Authorization via USSD Push

In the following countries the payment is authorization via a USSD menu

- Bolivia
- El Salvador
- Honduras
- Paraguay

This USSD menu is pushed to the customer's mobile phone and requests to validate the transaction by sending the Tigo MFS PIN code. The flow is shown below:



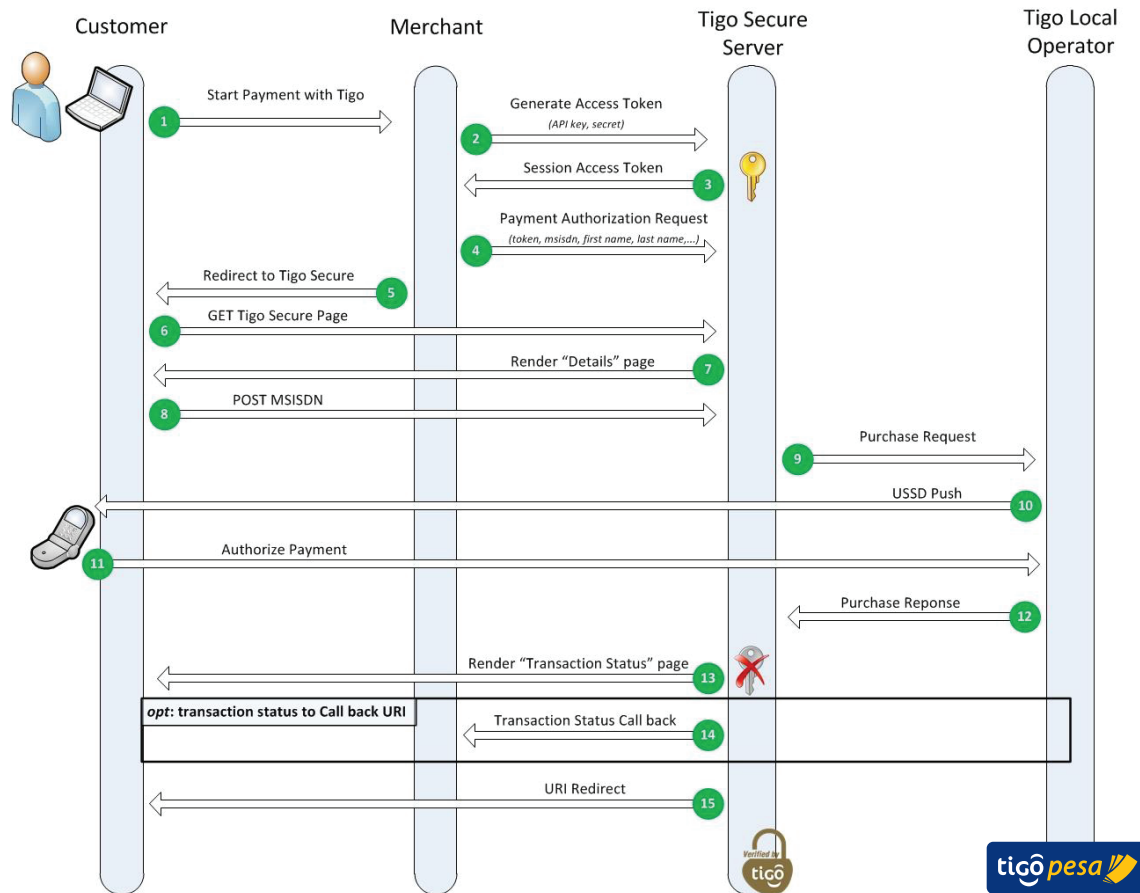


Figure 3-11: Payment Authorization flow USSD Push

- (1) The subscriber/customer initiates a Tigo MFS payment via the Merchant.
- (2-3) An Access token is requested for the Tigo Secure Server via the `GenerateAccessToken` service (Section 4.3.1) using the Apigee API Key and secret.
- (4-7) The Payment Authorization Request is made with the necessary payment details (Section 4.5.1) this will return a re-direct URL to the Tigo Secure Payment Authorization page to which the customer has to be redirected. The payment details page is shown in Figure 3-13.

**tigo pesa** Tigo Secure

First Name: John  
Last Name: Doe  
Email: johndoe@mail.com  
Tigo Mobile Number: 0981525272

\* mandatory

---

**Tigo Details**

Merchant	Amazon
Amount from merchant	USD 2.00
Fees	USD 0.50
Amount to be deducted from Tigo Money Account	PYG 11893.6

The foreign exchange rate that Tigo uses is 4677.44 from USD to PYG

**Confirm**

In case of problems or questions please contact your Tigo Cash Customer Care line \*111 from your tigo or +95 (021) 618 9000.

Figure 3-12: Payment details





- (8-10) The Customer submits the MISISDN and presses 'Confirm' to continue the transaction. A Purchase request is made which initiates a USSD session in which the Subscriber has to authorize the payment. The maximum duration is 5 minutes.

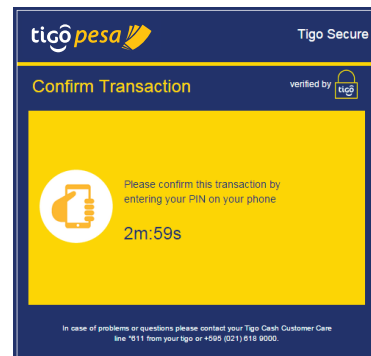


Figure 3-13: Pending Payment confirmation via USSD

- (11-13) The Customer authorizes the payment via USSD and the transaction status page is shown for limited duration.

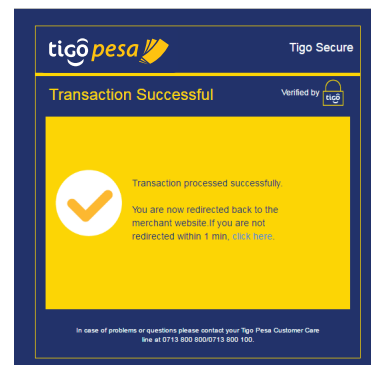


Figure 3-14: Payment result page

- (14) An optional callback URI is called with the final transaction status. This callback URI can be used in case the front-end server does not allow processing the financial transaction status.
- (15) The final redirect is done to the specified redirect URI.

### 3.8. Reverse Transaction

The Reverse Transaction Service can be used to reverse or refund a successful transaction made via the Online Payment.

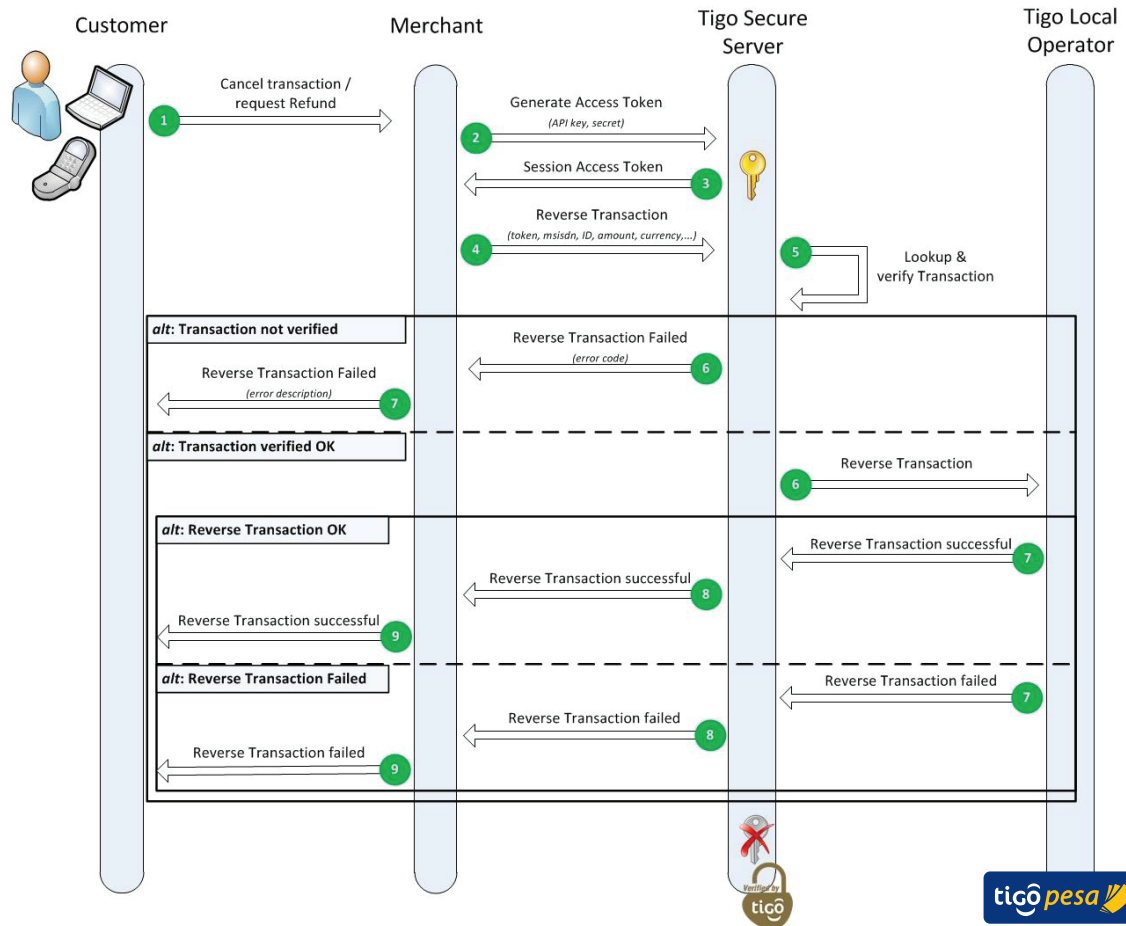


Figure 3-15: Reverse Transaction flow


## 4. API Specification

### 4.2. Introduction

This section covers the API specifications. Each service is divided in a Request and Response section containing the overview of the parameters and example requests and responses. The list of error codes is included in Annex A.6.4.

The following third-level domains are available for the Tigo Secure services:

<https://secur sandbox.tigo.com/> test environment  
<https://secure.tigo.com/> production environment



In the sections below the service URLs are relative to these two Tigo Secure domains. For example for a service called 'service1' the following URL is specified in the API specification:

`<domain>/v1/service1`

To use service1 on the test environment use the URL:  
<https://secur sandbox.tigo.com/v1/service1>

and on the production environment:  
<https://secure.tigo.com/v1/service1>

### 4.3. Generate Access Token Service

The Generate Access Token service is used to get a valid access token. The Millicom partner can only use the assigned API services via this access token. The access token has to be specified in each request header as described in the sub paragraphs below.

The validity of the access token is time limited and after each session the token is invalidated irrespective whether the service call resulted in a positive result or a failure. The expiry time is specified in the response.

#### 4.3.1. Request

Generate Access Token Request	
<b>URL</b>	<code>&lt;domain&gt;/v1/oauth/generate/accesstoken?grant_type=client_credentials</code>
<b>Method</b>	POST
<b>Headers</b>	Content-Type: application/x-www-form-urlencoded
<b>Body</b>	<code>client_id=&lt;client_id&gt;&amp;client_secret=&lt;client_secret&gt;</code>

with in the body:

- `<client_id>` is the unique client identifier as assigned during the registration process with Millicom
- `<client_secret>` is the secret/password as provided during the registration process with Millicom

### 4.3.2. Response

Average response time: < 1 second

Maximum response time: 5 seconds

In case as valid client\_id and client\_secret are submitted the following response is returned:

HTTP response code: 200 OK

JSON response body:

Parameter	Type	Description
accessToken	String	unique access token
issuedAt	Integer	Access Token issue Date and time as Unix time
expiresIn	Integer	Expiry time in seconds

Table 1: Generate Access Token Response parameters

Example response:

Response code: 200 OK

Response body:

```
{
  "accessToken": " ABcdef123456ABcdef123456ABcd",
  "issuedAt": "1410268728383",
  "expiresIn": "599",
}
```

In case incorrect client\_id or client\_secret are provided the following error is returned:

HTTP response code: 401 Unauthorized

JSON response body:

Parameter	Type	Description
ErrorCode	String	Error code
Error	String	Error description

Example response:

Response code: 401 Unauthorized

Response body:

```
{
  "ErrorCode" : "invalid_client",
  "Error" : "Client credentials are invalid"
}
```

### 4.4. System Status Service

The System Status Service is provided to monitor the health of the service periodically (heartbeat signal). The service returns the status of both the network connectivity and the application status to the Tigo Secure server and from the Tigo Secure Server to the Tigo Operations.

#### 4.4.1. Request

##### System Status Request

<b>URL</b>	<domain>/v1/tigo/systemstatus	
<b>Method</b>	GET	
<b>Header</b>	accessToken	<valid access token>

#### 4.4.2. Response

Average response time: < 1 second

Maximum response time: 5 seconds

HTTP response code: 200 OK

JSON Response body:

Parameter	#	Type	Description
tigoSecureStatusCode	1	Integer	Tigo Secure Server status code 0 = OK Any other number than 0 indicates a problem occurred
statusDescription	1	String	Description
TigoOperationStatus	0..n		
country	1	String	Three letter country code (ISO 3166-1 Annex A.2)
code	1	Integer	Tigo Secure Server status code 0 = OK Any other number than 0 indicates a problem occurred
description	1	String	Description

Table 2: System Status Response parameters

Example response:

```

{
  "tigoSecureStatusCode" : 0,
  "statusDescription" : "OK",
  "TigoOperationStatus" :
  {
    { "country": "TZA", "code": 0, "description": "OK" }
    { "country": "SEN", "code": 0, "description": "OK" }
  }
}

```

## 4.5. Payment Authorization service

#### 4.5.1. Request

For the Tigo Secure Online Payment Authorization a redirect to the following URL has to be done including a JSON request with the required payment details:

##### Payment Authorization Request

<b>URL</b>	<domain>/v1/tigo/payment-auth/authorize	
<b>Method</b>	POST	
<b>Header</b>	Content-Type	application/json
	accessToken	<valid access token>



## JSON Request body:

Parameter	#	Type	Description
MasterMerchant	1		
account	1	String	MFS Account number in the destination country (account to credit)
pin	1	String	MFS Account PIN code
id	1	String	Identifier of master merchant (i.e. company name) as provided by Millicom
Merchant	0..1		<i>[optional]</i>
reference	1	String	Reference of the originating merchant (company name) in case the payment was made on behalf of another company
fee	0..1	Decimal	Merchant fee for the transaction in the origin currency. This fee is charged from the merchant. Information about this fee will not be communicated to the subscriber. This information is confidential and is to be used for reconciliation
currencyCode	0..1	String	Currency code of the Merchant fee (see Annex A.1)
Subscriber	1		
account	1	String	MFS Account number (msisdn) of the paying subscriber (account to debit)
countryCode	1	String	Country code dialing prefix (annex A.4)
country	1	String	Three letter country code (ISO 3166-1 Annex A.2)
firstName	0..1	String	First name of the subscriber
lastName	0..1	String	Last name of the subscriber
emailId	0..1	String	<i>[optional]</i> Email address
redirectUri	1	String	Redirection URI to redirect after completing the payment
callbackUri	0..1	String	<i>[optional]</i> Result callback URI
language	1	String	Three letter code for the language (ISO 639-3 see Annex A.3)
terminalId	0..1	String	<i>[optional]</i> Terminal ID
originPayment	1		
amount	1	Decimal	Total amount in the currency of the original merchant payment
currencyCode	1	String	Currency code of the payment (see Annex A.1)
tax	1	Decimal	Tax for the transaction in the origin currency
fee	1	Decimal	Fee applied by the Master Merchant for the transaction in the origin currency. This fee is charged from the subscriber and will be shown to the subscriber. If no fee has been applied the field can be set to 0
exchangeRate	0..1	Decimal	<i>[optional]</i> Exchange rate between the origin currency (currency of the sending country) and local currency (currency of the receiving country)
LocalPayment	1		
amount	1	Decimal	Total amount in the local currency of the paying subscriber
currencyCode	1	String	Currency code of the MFS account of the paying subscriber (local currency)(see Annex A.1)
transactionRefId	1	String	Reference Identifier in order to uniquely identify the transaction.

**Table 3: Payment Authorization Request parameters**

**Sample Request:**

```
{
  "MasterMerchant":
  {
    "account": "255321321321",
    "pin": "1234",
    "id": "CompanyName"
  },
  "Merchant":
  {
    "reference": "Amazon",
    "fee": "23.45",
    "currencyCode": "EUR"
  },
  "Subscriber":
  {
    "account": "255111111111",
    "countryCode": "255",
    "country": "tza",
    "firstName": "John",
    "lastName": "Doe",
    "emailId": "johndoe@mail.com"
  },
  "redirectUri": "https://someapp.com/payment/redirecturi",
  "callbackUri": "https://someapp.com/payment/statuscallback",
  "language": "eng",
  "terminalId": "",
  "originPayment":
  {
    "amount": "75.00",
    "currencyCode": "USD",
    "tax": "0.00",
    "fee": "25.00"
  }
  "exchangeRate": "2182.23",
  "LocalPayment":
  {
    "amount": "218223.00",
    "currencyCode": "TZS"
  },
  "transactionRefId": "0ale39ab"
}
```



Make sure to **use the Access Token only once** to initiate a **Payment Authorization**. For each Payment Authorization request a new access token has to be generated. This is because the access token is invalidated after the transaction completed. Any other additional transaction initiated with the same access token will therefore fail.



Per Payment Authorization make sure to **use a unique transaction reference identifier** (transactionRefId) to identify the transaction. This will guarantee that the transaction is logged and traced correctly.

#### 4.5.2. Response

Average response time: < 1 second

Maximum response time: 5 seconds

HTTP response code: 200 OK

JSON Response body:

Parameter	#	Type	Description
transactionRefId	1	String	Unique reference Identifier of the transaction
redirectUrl	1	String	Tigo Secure redirect URL which has to be used to redirect the Customer to the correct Tigo Secure Payment Authorization webpage
authCode	1	String	Unique code to authenticate the transaction for the customer when redirecting
creationDateTime	1	String	Transaction Creation Date and Time

Table 4: Payment Authorization Response parameters

Example response:

```
{
  "transactionRefId": "0a1e39ab",
  "redirectUrl": "https://securesandbox.tigo.com/v1/payment_auth/transactions?...auth_code=123123123&transaction_ref_id=0a1e39ab&lang=eng",
  "authCode": "123123123",
  "creationDateTime": "Fri, 10 Oct 2014 13:58:25 UTC"
}
```

#### 4.5.3. Payment status callback

After the customer completes the payment via Tigo Secure the status is reported back. This is done via the optional callback URI or – in case this callback URI has not been specified – in the redirect URI as specified in the Payment Authorization Request. The optional callback URI will be called reporting back the transaction status with the following parameters:

##### Payment status Callback

<b>Method</b>	POST
<b>Headers</b>	Content-Type: application/x-www-form-urlencoded
<b>Body</b>	trans_status=<transaction status success/fail>&... transaction_ref_id=<transaction ref ID>&... external_ref_id=<external_ref_id>&... mfs_id=<mfs_id>&... verification_code=<Access Token>&... error_code=<error_code>

Parameter	Description
trans_status	Transaction status: success for a successful transaction fail in case of a failed transaction
transaction_ref_id	Transaction Reference Identifier as specified in the request
external_ref_id	[optional] Tigo transaction Id of the request and





<b>mfs_id</b>	responses between the internal servers. This will only be sent back in case of a successful payment <i>[optional]</i> MFS Platform transaction id of the payment. This will only be sent back in case of a successful payment
<b>verification_code</b>	<i>[optional]</i> The verification code is the invalidated Access Token as generated at the start of the payment authorization flow. This code has to be used to uniquely identify that payment status is reported back by Tigo Secure. Note that this access token is invalidated after the transaction failed/succeeded/expired so it can't be reused. The verification code (Access Token) <u>will be omitted</u> in case the transaction failed. This is to prevent that a malicious callback can be done with a modified transaction status.
<b>error_code</b>	<i>[optional]</i> The error code in case the transaction failed. The error codes are defined in Annex A.6.4.1.

**Table 5: Payment status callback parameters**

### Successful payment callback example:

```
POST HTTP/1.1
Host: <callback URI>
Content-Type: application/x-www-form-urlencoded
Cache-Control: no-cache

trans_status=success&transaction_ref_id=0a1e39ab
&external_ref_id=38c1069f-2497-4f9c-
9&mfs_id=CO140924.1414.A00113&verification_code=pfCIHgyWWg6qsUIOVFS
u2HR3F4jy&lang=eng
```





The verification code in the status callback is the invalidated Access Token as generated at the start of the payment transaction.

**In order to confirm that successful payment status has been reported back by Tigo Secure the following steps have to be performed:**

1. Lookup the payment transaction using the `transaction_ref_id`
2. Compare the `verification_code` against the original access token as used during the transaction
3. Only when the `verification_code` is equal to the original access token can the payment be treated as successful.

In case of a **mismatch** between the verification code and the access token the transaction should be **treated as failed** and reported back to Millicom. The `external_ref_id` and `transaction_id` can be used for traceability of the transaction within the Millicom Tigo Operation.

#### Failed payment example:

```
POST HTTP/1.1
Host: <callback URI>
Content-Type: application/x-www-form-urlencoded
Cache-Control: no-cache

trans_status=fail&transaction_ref_id=0a1e39ab-d0ec-4f8b-9746-
b2c4122220b2123&error_code=purchase-3008-30434-E>
```



For a failed transaction the verification code (access token) is not reported back. This is to prevent a malicious callback with a modified transaction status.

After the payment status callback a HTTP redirect will be done to the URI as specified in the `redirectUri` parameter in the Payment Authorization Request without any extra parameters.

Note that in case no `callbackUri` was specified in the original request the payment status is reported back in the `redirectUri` in the manner as for the callback URI explained above.



## 4.6. Validate MFS Account Service

The Validate MFS Account Service can be used to check whether the subscriber has a valid MFS account in the designated country. The request requires the subscriber MSISDN, first name and last name and country code as shown below.

### 4.6.1. Request

Validate MFS Account Request	
<b>URL</b>	<domain>/v1/tigo/mfs/validateMFSAccount
<b>Method</b>	POST
<b>Header</b>	Content-Type                      application/json
	accessToken                          <valid access token>

JSON Request body:

Parameter	Cardinality	Type	Description
transactionRefId	1	String	Reference Identifier in order to uniquely identify the transaction
ReceivingSubscriber	1		
— account	1	String	MFS Account to validate of the receiving subscriber
— countryCallingCode	1	Integer	County Calling code
— countryCode	1	String	Three letter country code (ISO 3166-1)
— firstName	0..1	String	<i>[optional]</i> First name of the subscriber
— lastName	0..1	String	<i>[optional]</i> Last name of the subscriber

Table 6: Validate MFS Account Request parameters

Example request:

```
{
  "transactionRefId" : "1300074238",
  "ReceivingSubscriber" :
  {
    "account" : "255658123964",
    "countryCallingCode" : "255",
    "countryCode" : "TZA",
    "firstName" : "John",
    "lastName" : "Doe"
  }
}
```



#### 4.6.2. Response

Average response time: 3 seconds

Maximum response time: 5 seconds

HTTP response code: 200 OK

JSON response body:

Parameter	Type	Description
ValidateMFSAccountResponse		
└ ResponseHeader		
└ GeneralResponse		
└ correlationID	String	The is the transaction id as sent in the request
└ status	String	Status of executing the account validation request (OK, ERROR)
└ code	String	Status code of the account validation (see Annex A.6.4)
└ description	String	Technical and brief description of the result
└ ResponseBody		
└ validMFSAccount	String	MFS account validation status <code>true</code> = account valid for provided details

Table 7: Validate MFS Account Response parameters

The following response is returned for a valid MFS account:

HTTP response code: 200 OK

JSON response body:

```

{
  "ValidateMFSAccountResponse":
  {
    "ResponseHeader":
    {
      "GeneralResponse":
      {
        "correlationID":1234,
        "status":"OK",
        "code":"Validatemfsaccount-3018-0000-S",
        "description":"Provided MSISDN is a valid
MFSAccount."
      }
    },
    "ResponseBody":
    {
      "validMFSAccount":"true"
    }
  }
}

```

In case of an invalid (non-existent) MFS account the follow response is returned:



HTTP response code: 500 Internal Server Error  
JSON response body:

```

{
  "Fault": {
    "faultcode": "env:Server",
    "faultstring": "Subscriber not found.",
    "detail": {
      "ValidateMFSAccountFault": {
        "ResponseHeader": {
          "GeneralResponse": {
            "correlationID": 1300074238,
            "status": "ERROR",
            "code": "Validatemfsaccount-3018-3001-E",
            "description": "Subscriber not found."
          }
        }
      }
    }
  }
}

```

#### 4.7. Deposit Remittance Service

With the Deposit Remittance Service the money for the international remittance is deposited in the subscriber's wallet. The partner wallet is debited and the subscriber wallet is credited for the amount in the local currency as specified in the request. The response will confirm success or failure of the money deposit which includes a unique Transaction ID from the MFS platform.

##### 4.7.1. Request

Deposit Remittance Request		
<b>URL</b>	<domain>/v1/tigo/mfs/depositRemittance	
<b>Method</b>	POST	
<b>Header</b>	Content-Type	application/json
	accessToken	<valid access token>

JSON request body:

Parameter	#	Type	Description
transactionRefId	1	String	Unique Transaction Reference Identifier
PaymentAggregator	1		
account	1	String	MFS Account number in the destination country
pin	1	String	MFS Account PIN code
id	1	String	Identifier of the payment aggregator (i.e. company name as provided by Millicom)
Sender	0..1		[optional]
firstName	1	String	First name of the Sender. This field can be left blank in case the information is not available.
lastName	1	String	Last name of the Sender. This field can be left blank in case the information is not available.
msisdn	0..1	String	[optional] MSISDN of the Sender
emailAddress	0..1	String	[optional] e-mail address of the Sender



Parameter	#	Type	Description
ReceivingSubscriber	1		
— account	1	String	MFS Account of the receiving subscriber
— countryCallingCode	0..1	Integer	[optional] Country Calling code
— countryCode	1	String	Three letter country code (ISO 3166-1 Annex A.2)
— firstName	1	String	First name of the subscriber
— lastName	1	String	Last name of the subscriber
OriginPayment	0..1		[optional]
— amount	1	Decimal	Total amount in the currency of the sending country
— currencyCode	1	String	Currency code of the sending country (see Annex A.1)
— tax	1	Decimal	Tax for the transaction in the origin currency
— fee	1	Decimal	Fee for the transaction in the origin currency
exchangeRate	0..1	Decimal	[optional] Exchange rate between the origin currency (currency of the sending country) and local currency (currency of the receiving country)
verificationRequest	0..1	Boolean	[optional] Verification flag ( <code>true/false</code> ). This feature is currently not supported. Only set to <code>false</code> otherwise the transaction will fail
sendTextMessage	0..1	Boolean	[optional] Flag to indicate whether a text message has to be sent ( <code>sendTextMessage = true</code> ) to the receiving subscriber in the following cases: Successful deposit: informing the subscriber received an international remittance with the amount, remittance partner and optionally the name of the sender. Unsuccessful deposit cause by the subscriber not signed up for a MFS account: informing the subscriber an international remittance was missed with the amount, remittance partner and optionally the name of the sender and the suggestion to open an MFS account.
LocalPayment	1		
— amount	1	Decimal	Total amount to payout in the local currency of the receiving subscriber (see Annex A.5 for formatting)
— currencyCode	1	String	Currency code of the receiving country (see Annex A.1)

Table 8: Deposit Remittance Request parameters

### Example Deposit Remittance Request

```

{
  "transactionRefId" : "1300074238",
  "PaymentAggregator" : {
    "account" : "255123123123",
    "pin" : "1234",
    "id" : "Company Name"
  },
  "Sender" : {
    "firstName" : "Jane",
    "lastName" : "Doe",
    "msisdn" : "2551234123423",
    "emailAddress" : "janedoe@mail.com"},
  "ReceivingSubscriber" : {
    "account" : "2551111111111",
    "countryCallingCode": "255",
    "countryCode" : "TZA",
    "firstName" : "John",
    "lastName" : "Doe"
  },
  "OriginPayment" : {
    "amount" : "100.00",
    "currencyCode" : "EUR",
    "tax" : "10.00",
    "fee" : "25.00"
  },
  "exchangeRate" : "2182.23",
  "verificationRequest" : "true",
  "sendTextMessage" : "true",
  "LocalPayment" : {
    "amount" : "200",
    "currencyCode" : "TZS"
  }
}

```

#### 4.7.2. Response

Average response time: 3 seconds

Maximum response time: 5 seconds

HTTP response code: 200 OK

JSON response body:

Parameter	Type	Description
DepositRemittanceResponse		
└ ResponseHeader		
└ GeneralResponse		
correlationID	String	The is the transaction id as sent in the request
status	String	Status of executing the account validation request (OK, ERROR)
code	String	Status code of the account validation (see codes below)
description	String	Technical and brief description of the result
└ ResponseBody		
transactionId	String	Transaction Identifier from the MFS Platform

Table 9: Deposit Remittance Response parameters



## Example Response

```

{
  "DepositRemittanceResponse":
  {
    "ResponseHeader":
    {
      "GeneralResponse":
      {
        "correlationID":1300074238,
        "status":"OK",
        "code":"depositremittance-3017-0000-S",
        "description":"The Transaction is completed
        successfully."
      }
    },
    "ResponseBody":
    {
      "transactionId":"CO140912.1700.A00059"
    }
  }
}
    
```

## 4.8. Reverse Transaction service

### 4.8.1. Request

Reverse Transaction Request

**URL** <domain>/v1/tigo/mfs/reverseTransaction

**Method** POST

**Header** accessToken <valid access token>

JSON request body:

Parameter	#	Type	Description
MasterAccount	1		
account	1	String	The MFS account of the Master Merchant / Payment Aggregator as used in the original request Payment Request or Deposit Remittance API request
pin	1	String	MFS Account PIN code for the Master Account
id	1	String	Identifier of Master Merchant (i.e. company name) as provided by Millicom
transactionRefId	1	String	Transaction Reference Identifier as submitted in the request (transactionRefId)
<b>mfsTransactionId</b>	1	String	The MFS Transaction Identifier for the transaction this maps to the Payment Authorization status callback mfs_id value or the DepositRemittanceResponse transactionId
countryCode	1	String	Three letter country code (ISO 3166-1 Annex A.2)
subscriberAccount	0..1	String	MFS Account of the subscriber (MSISDN)



Parameter	#	Type	Description
LocalPayment	0..1		Authorization or [optional]
└ amount	1	Decimal	Total amount of the transaction in the local currency (see Annex A.5 for the formatting)
└ currencyCode	1	String	Currency code of the Tigo country (see Annex A.1)

**Table 10: Reverse Transaction Request parameters**

Example request:

```

{
  "MasterAccount" :
  {
    "account" : "255321321321",
    "pin" : "1234",
    "id" : "CompanyName"
  },
  "transactionRefId" : "0a1e39ab",
  "mfsTransactionId" : "CO140924.1414.A00113",
  "countryCode" : "tza",
  "subscriberAccount" : "255111111111",
  "LocalPayment" :
  {
    "amount" : " 218223.00",
    "currencyCode" : "TZS"
  }
}

```

#### 4.8.2. Response

## 4.9. Payment Authorization Transaction Status service

### 4.9.1. Request

#### Get Transaction Status Request

<b>URL</b>	<domain>/v1/payment-auth/transactions/<MasterMerchant ID><transactionRefId >
<b>Method</b>	GET
<b>Header</b>	accessToken <valid access token>

with <MasterMerchant ID><transactionRefId> the MasterMerchant Identifier and transaction reference ID as specified in the Payment Authorization Request. For example when the below values were provided in the original Payment Authorization Request (Section 4.5.1):

```
"MasterMerchant":
{
...
  "id": "Company Name"
...
"transactionRefId": "0a1e39ab"
```

The example request to retrieve the payment authorization transaction status will be in that case:

```
GET /v1/payment-auth/transactions/Company Name0a1e39ab HTTP/1.1
Host: <host>
accessToken: <accessToken>
Cache-Control: no-cache
```

### 4.9.2. Response

Average response time: < 1 second

Maximum response time: 5 seconds

HTTP response code: 200 OK

JSON response body:

Parameter	#	Type	Description
<b>Transaction</b>			
refId	1	String	Unique Transaction Reference Identifier as provided in the initial request
externalRefId	0..1	String	[optional] Tigo transaction Id of the request and responses between the internal servers. This will only be returned for successful transactions
mfsId	0..1	String	[optional] MFS Platform transaction id of the payment. This will only be returned for successful transactions

Parameter	#	Type	Description
			in the following format: Fri, 10 Oct 2014 13:58:25 UTC
— Status	1	String	Transaction status: Success Fail
— completedOn	1		Completion date and time of the transaction in the following format: Fri, 10 Oct 2014 13:58:54 UTC
MasterMerchant	1		
— account	1	String	MFS Account number in the destination country (account to credit)
— id	1	String	Identifier of master merchant (i.e. company name)
Merchant	0..1		<i>[optional]</i>
— reference	1	String	Reference of the originating merchant (company name) in case the payment was made on behalf of another company
— fee	1	Decimal	Merchant fee for the transaction in the origin currency
— currencyCode	1	String	Currency code of the Merchant fee (see Annex A.1)
Subscriber	1		
— account	1	String	MFS Account number (msisdn) of the paying subscriber (account to debit).
— countryCode	1	String	Country code dialing prefix
— country	1	String	Three letter country code (ISO 3166-1 Annex A.2)
— firstName	1	String	First name of the subscriber
— lastName	1	String	Last name of the subscriber
— emailId	0..1	String	<i>[optional]</i> Email address
redirectUri	1	String	Redirection URI to redirect after completing the payment
callbackUri	0..1	String	<i>[optional]</i> Result callback URI
language	1	String	Three letter code for the language (ISO 639-3 see Annex 0)
terminalId	0..1	String	<i>[optional]</i> Terminal ID
OriginPayment	1		
— Amount	1	Decimal	Total amount in the currency of the sending country
— currencyCode	1	String	Currency code of the payment (see Annex A.1)
— tax	1	Decimal	Tax for the transaction in the origin currency
— fee	1	Decimal	Fee applied by the Master Merchant for the transaction in the origin currency
exchangeRate	0..1	Decimal	<i>[optional]</i> Exchange rate between the origin currency (currency of the sending country) and local currency (currency of the receiving country)
LocalPayment	1		
— amount	1	Decimal	Total amount in the local currency of the paying subscriber



Parameter	#	Type	Description
currencyCode	1	String	Currency code of the sending country (see Annex A.1)

**Table 11: Payment Authorization Transaction Status Response parameters**

Sample Payment Authorization Transaction Status response:

```

{
  "Transaction" :
  {
    "refId":"0ale39ab-d0ec-4f8b-9746-b2c4122220b2c120ww40",
    "externalRefId" : "38c1069f-2497-4f9c-9",
    "mfsId" : "CO140924.1414.A00113",
    "createdOn" : "Fri, 10 Oct 2014 13:58:25 UTC",
    "status" : "success",
    "completedOn" : "Fri, 10 Oct 2014 13:58:31 UTC",
  },
  "MasterMerchant":
  {
    "account":"255321321321",
    "id":"Skrill Ltd"
  },
  "Merchant":
  {
    "reference":"Acme, Inc",
    "fee":"23.45",
    "currencyCode":"TZS",
  }
  "Subscriber":{
    "account":"255111111111",
    "countryCode": "255",
    "country":"tza",
    "firstName":"John",
    "lastName":"Doe",
    "emailId" : "johndoe@mail.com"
  },
  "redirectUri":"https://someapp.com/payment/redirecturi",
  "callbackUri":" https://someapp.com/payment/statuscallback",
  "language":"eng",
  "terminalId":"","
  "originPayment":
  {
    "amount":"75.00",
    "currencyCode":"USD",
    "tax":"0.00",
    "fee":"25.00"
  },
  "exchangeRate":"2182.23",
  "LocalPayment":
  {
    "amount":"218223.00",
    "currencyCode":"TZS"
  }
}
    
```

## 4.10. Deposit Remittance Transaction Status service

### 4.10.1. Request

#### Get Deposit Remittance Transaction Status Request

<b>URL</b>	<domain>/v1/tigo/mfs/depositRemittance/transactions/<PaymentAggregator ID><Transaction ID>
<b>Method</b>	GET
<b>Header</b>	accessToken <valid access token>

with < PaymentAggregator ID>< Transaction ID> the Payment Aggregator Identifier and transaction reference ID as specified in the Deposit Remittance Request. For example when the below values were provided in the original Payment Authorization Request (Section 4.5.1):

```
{
  "transactionRefId" : "1300074",
  "PaymentAggregator" : {
    ...
    "id" : "Company Name"
    ...
  }
}
```

#### Example request:

```
GET /v1/tigo/mfs/depositRemittance/transactions/Company Name1300074
HTTP/1.1
Host: <host>
accessToken: <accessToken>
Cache-Control: no-cache
```

### 4.10.2. Response

Average response time: < 1 second  
Maximum response time: 3 seconds

HTTP response code: 200 OK  
JSON response body:

Parameter	#	Type	Description
Transaction	1		
refId		String	Unique Transaction Reference Identifier as provided in the initial request
status	1		Transaction status success/ fail
mfsId	1	String	MFS Platform transaction id of the payment
errorCode	0..1	String	Error code in case of a failed transaction
PaymentAggregator	1		
account	1	String	MFS Account number in the destination country
id	1	String	Identifier of the payment aggregator (i.e.

Sender	0..1		
firstName	1	String	First name of the Sender
lastName	1	String	Last name of the Sender
msisdn	0..1	String	<i>[optional]</i> MSISDN of the Sender
emailAddress	0..1	String	<i>[optional]</i> e-mail address of the Sender
ReceivingSubscriber	1		
account	1	String	MFS Account of the receiving subscriber
countryCallingCode	0..1	Integer	<i>[optional]</i> Country Calling code
countryCode	1	String	Three letter country code (ISO 3166-1 Annex A.2)
firstName	1	String	First name of the subscriber
lastName	1	String	Last name of the subscriber
OriginPayment	0..1		<i>[optional]</i>
amount	1	Decimal	Total amount in the currency of the sending country
currencyCode	1	String	Currency code of the sending country (see Annex A.1)
tax	1	Decimal	Tax for the transaction in the origin currency
fee	1	Decimal	Fee for the transaction in the origin currency
exchangeRate	0..1	Decimal	<i>[optional]</i> Exchange rate origin payment currency and local payment currency
verificationRequest	0..1	Boolean	<i>[optional]</i> currently not used
sendTextMessage	0..1	Boolean	<i>[optional]</i> Flag to send text message after complete the transaction
localPayment	1		
amount	1	Decimal	Total amount to payout in the local currency of the receiving subscriber
currencyCode	1	String	Currency code of the receiving country (see Annex A.1)

**Table 12: Deposit Remittance Transaction Status Response parameters**

### Example Response:

```
{
  "Transaction": {
    "refId": "1300074239",
    "status": "success",
    "mfsId": "CI141127.2125.A03951"
  },
  "PaymentAggregator" :
  {
    "account" : "255123123123",
    "id" : "CompanyName"
  },
  "Sender" :
  {
    "firstName" : "Jane",
    "lastName" : "Doe",
    "msisdn" : "441512121212",
    "emailAddress" : "janedoe@mail.com"
  },
  "ReceivingSubscriber" :
  {
    "account" : "255111111111",
    "countryCallingCode" : "255",
    "countryCode" : "TZA",
    "firstName" : "John",
    "lastName" : "Doe"
  },
  "OriginPayment" :
  {
    "amount" : "100.00",
    "currencyCode" : "EUR",
    "tax" : "10.00",
    "fee" : "25.00"
  },
  "exchangeRate" : "2182.23",
  "verificationRequest" : "false",
  "sendTextMessage" : "true",
  "localPayment" :
  {
    "amount" : "5555",
    "currencyCode" : "TZS"
  }
}
```



## A. Appendices

### A.1. Currency Codes

The Millicom supported currency codes are according to the ISO 4217 standard.

Table 13: Currency codes

Code	Currency
BOB	Boliviano
CDF	Congolese franc
COP	Colombian peso
EUR	Euro
GHS	Ghanaian cedi
GTQ	Guatemalan quetzal
PYG	Paraguayan guaraní
RWF	Rwandan franc
TZS	Tanzanian shilling
USD	United States dollar
XAF	CFA franc BEAC
XOF	CFA Franc



## A.2. Country Codes

The Millicom supported country codes are according to the ISO 3166-1 alpha-3 standard.

Table 14: Country letter codes

Code	Country
BOL	Bolivia, Plurinational State of
COD	Congo, the Democratic Republic of the
COL	Colombia
GHA	Ghana
GTM	Guatemala
HND	Honduras
PRY	Paraguay
RWA	Rwanda
SEN	Senegal
SLV	El Salvador
TCD	Chad
TZA	Tanzania, United Republic of



### A.3. Language Codes

The Millicom supported language codes are according to the ISO 639-3 standard.

Table 15: Language codes

Code	Language
<b>ara</b>	Arabic
<b>aym</b>	Aymara
<b>cab</b>	Garifuna
<b>emk</b>	Eastern Maninkakan
<b>eng</b>	English
<b>fra</b>	French
<b>ful</b>	Fulah
<b>grn</b>	Guarani
<b>jod</b>	Wojenaka
<b>jud</b>	Worodougou
<b>kfo</b>	Koro (Côte d'Ivoire)
<b>kga</b>	Koyaga
<b>kin</b>	Kinyarwanda
<b>lin</b>	Lingala
<b>lua</b>	Luba-Lulua
<b>miq</b>	Mískito
<b>mku</b>	Konyanka Maninka
<b>msc</b>	Sankaran Maninka
<b>mxx</b>	Mahou
<b>mzj</b>	Manya
<b>que</b>	Quechua
<b>snk</b>	Soninke
<b>spa</b>	Spanish
<b>srr</b>	Serer
<b>swa</b>	Swahili (macrolanguage)
<b>wol</b>	Wolof

## A.4. Country Calling Codes

Table 16: Country Calling Codes

Country	Calling code
Bolivia, Plurinational State of	591
Congo, the Democratic Republic of the	243
Colombia	57
Ghana	233
Guatemala	502
Honduras	504
Paraguay	595
Rwanda	250
Senegal	221
El Salvador	503
Chad	235
Tanzania, United Republic of	255



## A.5. Amount Format Convention

The amounts in the API requests are formatted according to the following standard:

#####.##

Whereby . (dot) will be used as the separator character for decimals (cents). No separator character is required (allowed) for thousands.

## A.6. Result and error codes

### A.6.1. HTTP status codes

The supported HTTP Status codes are shown in the table below.

Code	Message	Type	Description
200	OK	Success	Returned for all Successful Responses
400	Invalid Request	Error	One or more of the input mandatory fields are missing. Display an Invalid Request screen.
401	Unauthorized	Error	API Key is not valid
403	Forbidden	Error	API Key in not subscribed to service
405	Method Not Allowed	Error	Served for Methods other than POST
406	Not Acceptable	Error	Accept Header does not comply with x-www-form-urlencoded
500	Internal Server Error	Error	Any Error that is returned from the back-end systems (see next subparagraphs) or errors that are not covered by this list
505	HTTP Version Not Supported	Error	For Requests not on HTTP/1.1 protocol

### A.6.2. General error codes

#### A.6.2.1. IP address not whitelisted

The response below will be returned in case the IP address has not been whitelisted for the service called. Contact Millicom in order to add the correct IP address to the required services.

Status code: HTTP/1.1 403 Forbidden

```
{
  "fault":
  {
    "faultstring": "Access Denied for client ip : <IP address>",
    "detail":
    {
      "errorcode": "accesscontrol.IPDeniedAccess"
    }
  }
}
```

#### A.6.2.2. Invalid Request

In case the header or JSON request body is incorrect the error as shown below will be returned. Please check the request is

Status code: HTTP/1.1 400 Bad Request

```
{
  ErrorCode: "invalid_request"
  Error: "Missing required parameter transactionRefId"
}
```

#### A.6.2.3. Invalid Access Token

The following error is returned in case an invalid Access Token is provided

Status code: HTTP/1.1 401 Unauthorized

```
Invalid accessToken. Please enter valid token.
```

#### A.6.2.4. Access Token Expired

The following error is returned in case an invalid Access Token is provided

Status code: HTTP/1.1 401 Unauthorized

```
Expired accessToken. Please enter valid token.
```

#### A.6.2.5. Transaction Reference ID already used

The error below is returned in case Transaction Reference ID has already been used for a previous transaction.

Status code: HTTP/1.1 400 Invalid Request

```
{
  "errorCode": "invalid_request",
  "error": "transactionRefId already exists"
}
```

#### A.6.3. API Permission error

The error below is returned in case the API has not been activated for the given Client\_id and secret.

Status code: HTTP/1.1 401 Invalid Request

```
You don't have permission to access ... API. Please contact Millicom admin
```

#### A.6.4. Service specific result and error codes

The subsections below list the Result and Error codes per service. The nominal (successful) cases are covered in Section 4.

In case of an error a JSON response is returned with the following structure:



```
{
  "Fault":
  {
    "faultcode": "env:Server",
    "faultstring": "<error description>",
    "detail":
    {
      "ValidateMFSAccountFault":
      {
        "ResponseHeader":
        {
          "GeneralResponse":
          {
            "correlationID": <Tigo correlation ID>,
            "status": "ERROR",
            "code": "<error code>",
            "description": "<error description>"
          }
        }
      }
    }
  }
}
```

The error code provide in the “code” field has the following format:

<API>-<APIID>-<ERROR CODE>-<TYPE OF FAULT>, where

API – Tigo API name

APIID – unique numeric identifier of the API within Tigo

Error code – based on the range and the fault type .

Type of fault with:

E – business fault – 3000 range

F – fatal errors (such as network related faults) 2000 range

V – validation fault –4000 range

S – Success –0000 range

W – warning (scenarios such as partial responses) – 6000 range



#### A.6.4.1. Payment Authorization Service result codes

Code	Description
purchase-3008-0000-S	Successful Payment

Error code	Description	Error Category
purchase-3008-2501-F	Backend system error	Backend Error caused the transaction to fail.
purchase-3008-2502-F	Transaction timed out	The transaction timed out causing it to fail.
purchase-3008-3011-E	Unable to complete transaction invalid amount	Unable to complete transaction as amount is invalid
purchase-3008-3043-E	Transaction not authorize	The customer did not authorize the payment and therefore the transaction failed. This could be caused by the customer not confirming payment, incorrect verification code or PIN code, insufficient balance etc.
purchase-3008-3045-E	Cancel Transaction	The customer doesn't wish to complete the transaction and wants to cancel the transaction at its current state

#### A.6.4.2. Validate MFS Account Service result codes

Code	Description
validatemfsaccount-3018-0000-S	Provided MSISDN is a valid MFS Account.

Error code	Description	Error Category
------------	-------------	----------------





<b>validatemfsaccount-3018-4501-V</b>	Invalid Request. Please check the input and resubmit	OSB Validation Error
<b>validatemfsaccount-3018-3001-E</b>	<Backend error description>	Backend Error
<b>validatemfsaccount-3018-2501-F</b>	One or more back ends may be down. Please try again later.	Connection Error.
<b>validatemfsaccount-3018-2502-F</b>	Service call has timed out. Please try again later.	Timeout error.
<b>validatemfsaccount-3018-2505-F</b>	Service Authentication Failed.	OWSM Authentication Failure.
<b>validatemfsaccount-3018-2506-F</b>	Consumer is not authorized to use this service.	OWSM Authentication Failure.
<b>validatemfsaccount-3018-3603-E</b>	Internal service error has occurred.	Internal service error.
<b>validatemfsaccount-3018-3999-E</b>	Unknown/Uncaught error has occurred.	Unknown/Uncaught error has occurred.
<b>validatemfsaccount-3018-4502-V</b>	Invalid ISD code passed in the MSISDN	Validation Error
<b>validatemfsaccount-3018-4503-V</b>	Web Service Implementation is not available for this country	Validation Error
<b>validatemfsaccount-3018-4504-V</b>	Required additional parameters are not passed in the request.	When the required additional parameters are not passed
<b>validatemfsaccount-3018-4505-V</b>	Duplicate additional parameters passed in the request.	When the required additional parameters are repeated
<b>validatemfsaccount-3018-4506-V</b>	Invalid consumerId passed in the request.	When the consuming application does not send the Id , which helps middleware to identify the request originating request.



### A.6.4.3. Deposit Remittance Service result codes

Code	Description
<b>depositremittance-3017-0000-S</b>	The Transaction is completed successfully

Error Code	Description	Error Category
<b>depositremittance-3017-2501-F</b>	One or more back ends may be down. Please try again later.	Connection Error. This is a Tigo internal error and should be treated as 'Service not available'
<b>depositremittance-3017-2502-F</b>	Service call has timed out. Please try again later.	Timeout error
<b>depositremittance-3017-2505-F</b>	Service Authentication Failed.	OWSM Authentication Failure. This is a Tigo internal error and should be treated as 'Service not available'
<b>depositremittance-3017-2506-F</b>	Consumer is not authorized to use this service.	OWSM Authentication Failure. This is a Tigo internal error and should be treated as 'Service not available'
<b>depositremittance-3017-3001-E</b>	<Backend error description>	Uncaught error from the Tigo MFS Platform. This error should be treated as 'Service not available'
<b>depositremittance-3017-3002-E</b>	Authorization failed	The authorization for the transaction with the provided MFS Account details failed. Check the account details and send a new request. If the error persists contact Tigo to resolve
<b>depositremittance-3017-3003-E</b>	Password expired	The PIN/Password for the MFS account expired. Contact Tigo to reset the PIN
<b>depositremittance-3017-3004-E</b>	Sender account suspended	The MFS Account has been suspended. Contact Tigo to resolve
<b>depositremittance-3017-3005-E</b>	Sender account does not exist	The provided MFS Account does not exist on the MFS platform in the country. Check the MFS Account details and send a new transaction. If the error persists contact Tigo to resolve
<b>depositremittance-3017-3006-E</b>	Receiver Account suspended	The receiving MFS account has been suspended and therefore a remittance is not possible to this account.
<b>depositremittance-3017-3007-E</b>	Receiver Account does not exist	The receiving MFS account does not exist and therefore a



<b>depositremittance-3017-3008-E</b>	Invalid amount specified	remittance is not possible to this account. An invalid amount has been specified in the request. Send a new transaction with a correct amount.
<b>depositremittance-3017-3009-E</b>	Maximum balance threshold for receiver exceeded	The balance of the receiving MFS account has been exceeded and therefore a remittance is not possible to this account.
<b>depositremittance-3017-3010-E</b>	Maximum number of transaction for receiver account reached	The maximum number of transactions of the receiving MFS account has been reached and therefore a remittance is not possible to this account.
<b>depositremittance-3017-3011-E</b>	Maximum number of transaction for sender account reached	The maximum number of transactions of the sending MFS account has been reached and therefore a remittance is not possible to this account.
<b>depositremittance-3017-3012-E</b>	Transaction amount is less than the minimum transaction limit	The amount as passed in the request is less than the minimum transaction limit.
<b>depositremittance-3017-3013-E</b>	Maximum transaction limit exceeded	The amount as passed in the request is more than the maximum transaction limit.
<b>depositremittance-3017-3014-E</b>	Sender and receiver account are the same	Then sender and receiver accounts as specified in the request as the same.
<b>depositremittance-3017-3015-E</b>	Service timeout	The request to deposit the remittance has timed out and therefore has not been completed.
<b>depositremittance-3017-3016-E</b>	Insufficient funds	The account used for sending funds has insufficient funds. In case of the Payment aggregator account this should not occur. If it does occur Millicom has to be contacted.
<b>depositremittance-3017-3017-E</b>	Insufficient account permission	The account has insufficient permission to perform the deposit remittance. In case of the Payment aggregator account this should not occur and if it does Millicom has to be contacted.
<b>depositremittance-3017-3018-E</b>	User not found	The account specified in the request cannot be found.
<b>depositremittance-3017-3603-E</b>	Internal service error has occurred.	Internal service error. This is a Tigo internal error and should be treated as 'Service not available'
<b>depositremittance-3017-3999-E</b>	Unknown/Uncaught error has occurred.	Unknown/Uncaught error has occurred. This is a Tigo internal error and should be treated as 'Service not available'
<b>depositremittance-3017-4002-V</b>	Invalid amount passed in the request.	When the amount is less than or equal to zero



<b>depositremittance-3017-4501-V</b>	Invalid Request. Please check the input and resubmit	OSB Validation Error. This is a Tigo internal error and should be treated 'Service not available'
<b>depositremittance-3017-4502-V</b>	Invalid ISD code passed in the MSISDN	Validation Error. An incorrect MSISDN was sent. Send a new request with a correct MSISDN
<b>depositremittance-3017-4503-V</b>	Web Service Implementation is not available for this country	Validation Error. The selected country does not implemented the web service
<b>depositremittance-3017-4504-V</b>	Required additional parameters are not passed in the request.	This is a Tigo internal error and should be treated as 'Service not available'
<b>depositremittance-3017-4505-V</b>	Duplicate additional parameters passed in the request.	When the required additional parameters are repeated
<b>depositremittance-3017-4506-V</b>	Invalid consumerId passed in the request.	When the consuming application does not send the Id , which helps middleware to identify the request originating request



tigo *pesa* 